In the Claims:

Please replace claims 8 and 10, and add new claim 18, all as shown below.

1 – 3. (Cancelled)

4. (Previously Presented): The method of claim 17 wherein the superclass includes logic to handle server side tasks.

5. (Previously Presented): The method of claim 17 wherein the wrapper class is generated in bytecode.

6. (Original): The method of claim 5 wherein bytecode is generated for vendor methods not implemented in the superclass.

7. (Canceled)

8. (Currently Amended): The method of claim 17 wherein providing the wrapper object to an application~~,~~ allows the application to access standard features and non-standard vendor extensions.

9. (Canceled)

10. (Currently Amended): A method for processing an invocation using a dynamically generated wrapper, comprising:

receiving an invocation by a wrapper object, the wrapper object instantiated from a wrapper class, the wrapper class extended from a superclass which implements Java Database Connectivity, Java Message Service and Java Connector Architecture, the invocation directed to a wrapped resource adapter by an application;

initiating pre-processing by the wrapper object;

calling the wrapped resource adapter by the wrapper object;

receiving a result from the wrapped resource adapter by the wrapper object;

initiating post-processing by the wrapper object; and

provide the result to the application program.

11. (Original): The method of claim 10 wherein the pre-processing including calling a pre-invocation handler.

12. (Original): The method of claim 10 wherein the pre-invocation handler is configured to execute server-side code.

13. (Previously Presented): The method of claim 12 wherein the server-side code includes transaction processing code.

14. (Original): The method of claim 10 wherein post-processing including calling a post-invocation handler.

15. (Original): The method of claim 14 wherein the post-invocation handler is configured to perform post-processing server side tasks.

16. (Previously Presented): The method of claim 15 wherein the post-processing server-side tasks include transaction management.

17. (Previously Presented): A method for dynamically generating a wrapper object, comprising:

receiving a resource adapter class at an application server;

performing reflection on the resource adapter class to identify interfaces implemented by the resource adapter class;

dynamically generating a wrapper class at runtime that extends from a superclass, wherein the superclass implements Java Database Connectivity, Java Messaging Service, or Java Connector Architecture interfaces, and the wrapper class implements the interfaces identified through reflection;

instantiating a wrapper object from the wrapper class; and

providing the wrapper object to an application that requires support for the interfaces implemented by the resource adapter class.

18. (New): The method of claim 17 wherein the superclass is statically predefined.